Lustre in WAN Environment and Development

T. Stibor, B. Alborea, S. Haller, C. Huhn, D. Klein, B. Neuburger, M. Pausch, V. Penso, C. Preuß, T. Roth, W. Schön, and J. Trautmann
GSI, Darmstadt, Germany

Lustre in WAN Environment

Recently, Lustre has been started to be employed in wide area networks (WAN) to enable efficient data access beyond local HPC and cluster environments [2]. In WAN environments challenging problems such as reliable tera-link connections and security related issues such as authentication and data privacy/integrity [1] need to be addressed. For addressing the first issue a 120 GBit/s high-speed connection between GSI and the LoeweCSC based on LNET routers was implemented (see Fig. 1). This is the first step for the general tera scale project in the context of FAIR enabling partnered research institutes Lustre access for processing data.

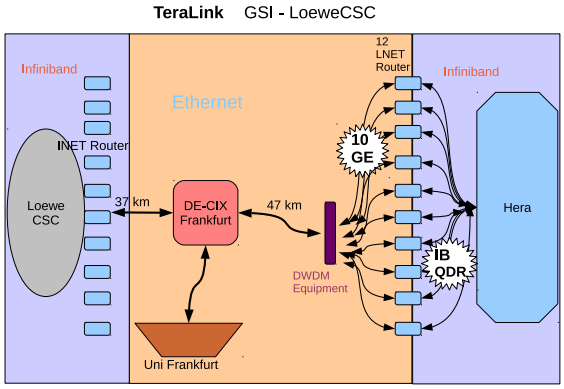

Figure 1: The high-speed connection is realized by bundling 12 machines equipped with 10 GBit/s ethernet cards acting as LNET routers. For seamless connecting both sides efficiently, Infiniband over IP is employed and network bandwidth saturation in experiments is achieved.

In the subsequent Section we briefly summarize the HPC development of a lightweight access control mechanism for addressing security related problems of Lustre in WAN environments.

A Lightweight Access Control Mechanism for Lustre in WAN Domains

For controlling access to Lustre of clients outside the GSI domain, a Linux kernel module based on Linux UID/GID and Lustre network identifier is developed. It allows to control *read* and *write* access on Lustre mount points comparable to Lustre's root squash functionality, however, for arbitrary specified UID's/GID's and Lustre network identifier ranges such as UID:$[1001 \ldots 2500]$,GID:$[2001 \ldots 3500]$, etc. In the context of wide area Lustre deployment the proposed mechanism enables a straightforward and lightweight access control of Lustre clients located in different wide area domains. The access control mechanism is implemented as a separate Linux kernel module and exports an access-granting function which is hooked into MDT system calls such `mdt_reint_open(...)` or `mdt_md_create(...)` to grant or deny access. For defining and removing rules, the Linux procfs is employed (see examples below).

Load kernel module & set rules:

```
>insmod ./lugac.ko
GSI Lustre UID/GID access control module v0.3 loaded
>echo "192.168.[67-70].[1-16]@tcp0 [500-600] 1012" > /proc/lugac
>echo "10.10.1.1@tcp5 [100-200] [100-200]" > /proc/lugac
```

Enforce rules:

```
user1@10.10.1.1:uid:5:gid:5>ls /lustre
ls:  cannot access /lustre:  Permission denied
user2@10.10.1.1:uid:105:gid:105>ls /lustre
data.raw
```

For more details and for studying the kernel module code see http://www.stibor.net/lugac.

Contributing Code to Lustre

The HPC group is now an active member of contributing and reviewing code for the Lustre file system, especially in the area of Kerberos and GSS. This encompass patches for latest Linux kernels as well as fixes for Kerberos and GSS (see http://git.whamcloud.com).

Outlook

The next project steps will encompass implementing a *kerberized* Lustre for $3.X$ Linux kernels to enable strong user authentication and data encryption/integrity. In addition, preliminary explorations and efficiency experiments on ZFS as a backbone file system for Lustre are performed.

References

[1] Josephine Palencia, Robert Budden, and Kevin Sullivan. Kerberized lustre 2.0 over the wan. In *Proceedings of the 2010 TeraGrid Conference*, pages 1–5. ACM Press, 2010.

[2] Stephen C. Simms, Gregory G. Pike, and Doug Balog. Wide area filesystem performance using lustre on the teragrid. In *Proceedings of the TeraGrid 2007 Conference*, 2007.